Complexity Insights of the Minimum Duplication Problem

Guillaume Blin, Paola Bonizzoni, Riccardo Dondi, Romeo Rizzi, Florian Sikora

Complexity Insights of the Minimum Duplication Problem

Guillaume Blin[a], Paola Bonizzoni[b], Riccardo Dondi[c], Romeo Rizzi[d], Florian Sikora[e]

[a] *Université Paris-Est, LIGM, UMR 8049 - France,* gblin@univ-mlv.fr
[b] *DISCo, Universitá degli Studi di Milano-Bicocca - Milano, Italy,* bonizzoni@disco.unimib.it
[c] *Dipartimento di Scienze umane e sociali, Universitá degli Studi di Bergamo, - Bergamo, Italy,* riccardo.dondi@unibg.it
[d] *Department of Computer Science, University of Verona - Verona, Italy,* Romeo.Rizzi@univr.it
[e] *PSL, Université Paris-Dauphine, LAMSADE, UMR 7243 - Paris, France,* florian.sikora@dauphine.fr

Abstract

The Minimum Duplication problem is a well-known problem in phylogenetics and comparative genomics. Given a set of gene trees, the Minimum Duplication problem asks for a species tree that induces the minimum number of gene duplications in the input gene trees. Recently, a variant of the Minimum Duplication problem, called Minimum Duplication Bipartite, has been introduced, where the goal is to find all *pre-duplications*, that is duplications that in the evolution precede the first speciation with respect to a species tree. In this paper, we investigate the complexity of both Minimum Duplication and Minimum Duplication Bipartite. First of all, we prove that the Minimum Duplication problem is APX-hard, even when the input consists of five uniquely leaf-labelled gene trees (improving upon known results on the complexity of the problem). Then, we show that the Minimum Duplication Bipartite problem can be solved efficiently with a randomized algorithm when the input gene trees have bounded depth. An extended abstract of this paper appeared in SOFSEM 2012 [1].

Keywords: Minimum Duplication Problem, Comparative Genomics, Computational Complexity, APX-hardness, Randomized algorithm

1. Introduction

The evolutionary history of the genomes of eukaryotes is the result of a series of evolutionary events, called *speciations*, that produce new species starting from a common ancestor. This evolutionary history has been deeply studied in computational biology, and it is usually represented using a phylogenetic tree called *species tree* [2]. A *species tree* is a rooted binary tree whose leaves are uniquely labelled by a set Λ representing the extant species, where the common ancestor of the contemporary species is associated with the root of the tree. The internal nodes represent hypothetical ancestral species (and the associated speciations). Speciations are not the only events that influence the evolution. Indeed, there are other events, such as gene duplications, gene losses and lateral gene transfers that, although not leading to new species, are fundamental in the evolution. In this paper we focus on gene duplications which are known to be essential for the evolution of many eukaryotes groups, such as vertebrates, insects and plants [3]. A gene duplication can be described as the genomic event that causes a gene inside a genome to be copied, resulting in two copies of the same gene that can evolve independently. Genes of extant species are called *homologous* if they evolved from a common ancestor through speciations and duplications events [4]. The evolution of homologous genes, with regards to the extant species, is usually represented using another special kind of phylogenetic tree, called *gene tree*. A *gene tree* is a rooted binary tree whose leaves are (not necessarily uniquely) labelled by elements of the set Λ. Despite the fact that biologically speaking leaves in the gene tree represent genes, for simplification, the gene tree is labelled according to the species from which the corresponding gene was sampled. Therefore, leaves similarly labelled represent duplicated genes that evolved independently and appear in a common extant species. As in the species tree, the root and the internal nodes respectively represent the common ancestor and ancestral genes explaining their evolution.

With regards to the set of labels Λ, gene and species trees are said to be *comparable*. Nevertheless, due to complex evolutionary processes, such as gene duplications and losses, comparable gene trees and species trees very often present incompatibilities. An interesting problem is then to reconcile the gene trees and species trees with hypothetical gene duplications. For example, in Fig. 1, given a comparable gene tree and species tree inducing incompatibilities, one can infer a reconciled tree based on the *a priori* duplication of gene g_1 into genes h and g_3 (h is a hypothetical ancestor of genes g_2,

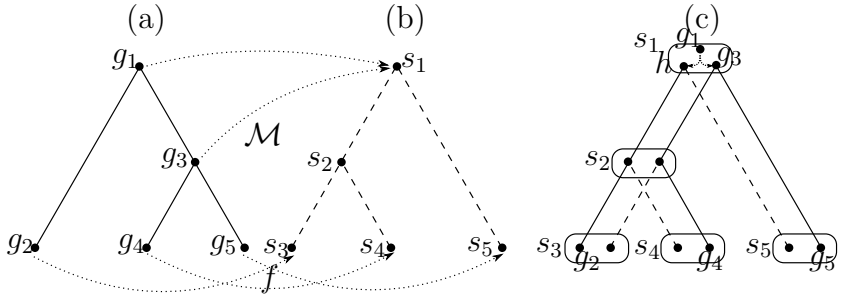

Figure 1: (a) a gene tree T. (b) a species tree S where $\mathcal{M}$ is the `lca` mapping from T to S; each gene in $\{g_2, g_4, g_5\}$ is mapped by function f in the species that gene belongs to. Nodes of S are labeled with s's. (c) a reconciled tree for T and S based on the a priori duplication of gene g_1 into genes h and g_3.

g_4), which afterwards both speciate according to the topology of the species tree.

Reconciliation is a widely-investigated problem, and different approaches have been proposed in the past based on the duplication-loss model [5, 6, 7, 8, 9, 10, 11, 12, 13, 14] and also extended to consider later gene transfer [15, 16, 17, 18, 19]. Some approaches are based on a probabilistic model that aims to infer how a gene tree evolves within a given species tree [5, 6, 17].

Based on the principle of parsimony, one is interested in finding the minimum number of gene evolutionary events that can explain all the incompatibilities. Notice that, while we focus on minimizing duplications, other possible costs have been considered, for example the minimization of losses or the minimization of duplications and losses [12, 9, 20].

This last can be inferred by the so-called *lowest common ancestor mapping* (`lca` mapping), denoted by $\mathcal{M}$. $\mathcal{M}$ maps each ancestral gene g of the gene tree to the most recent common ancestor of the extant species from which all the descendants of g were sampled. Given $\mathcal{M}$, a gene in the gene tree is a gene duplication if it has a descendant with the same $\mathcal{M}$ mapping. Then, the reconciliation cost is defined as the number of gene duplications in the gene tree induced by the species tree. Computing a specie tree inducing the minimum cost for this distance has been widely investigated under the name of the Minimum Duplication problem [21, 12, 20, 22] (defined formally afterwards).

1.1. Known results

The Minimum Duplication problem is known to be NP-hard [12]. Recently, the Minimum Duplication problem has been related to the Minimum Triplets Consistency problem [22], a problem known to be W[2]-hard [23] and not approximable within factor $O(\log n)$ [23]. These results coupled with the reduction provided in [22] implies that the Minimum Duplication problem is NP-hard, W[2]-hard (despite of [21]) and cannot be approximated within factor $O(2^{\log^{1-\varepsilon} n})$, even in the specific case of a forest composed of uniquely leaf-labelled gene trees with three leaves [22, 24] (notice that if the forest consists of a constant number of uniquely leaf-labelled gene trees with three leaves, then the problem is trivially in P).

Therefore, different heuristics and Integer Linear Programs have been developed [25, 26, 9, 27].

Recently, the Minimum Duplication Bipartite problem has been introduced to tackle the Minimum Duplication problem [28]. The Minimum Duplication Bipartite problem aims to find all the *pre-duplications*, that is duplications that in the evolution precede the first speciation with respect to a species tree (see Fig. 2 for an example). Roughly, this means that only the first level of the species tree is considered. Indeed, one is interested in knowing if a given species belongs to the subtree of S rooted at the left child of the root or at the right one. Therefore, one can view the species tree as a bipartition (Λ_1, Λ_2) of the set of species Λ. Solving the Minimum Duplication Bipartite problem recursively produces a natural greedy heuristic for the Minimum Duplication problem. The Minimum Duplication Bipartite problem was shown to be 2-approximable [28], but its complexity remains open.

In this contribution, we provide results for both the Minimum Duplication problem and the Minimum Duplication Bipartite problem. First of all, we prove that the Minimum Duplication problem is APX-hard, even when the input consists of five uniquely leaf-labelled gene trees (that is for a constant number of gene trees). Then, we show that the Minimum Duplication Bipartite problem can be solved efficiently with a randomized algorithm when the input gene trees have bounded depth.

2. Preliminaries

In this section we introduce some preliminary definitions and properties that will be useful in the rest of the paper. Consider a binary tree U, we

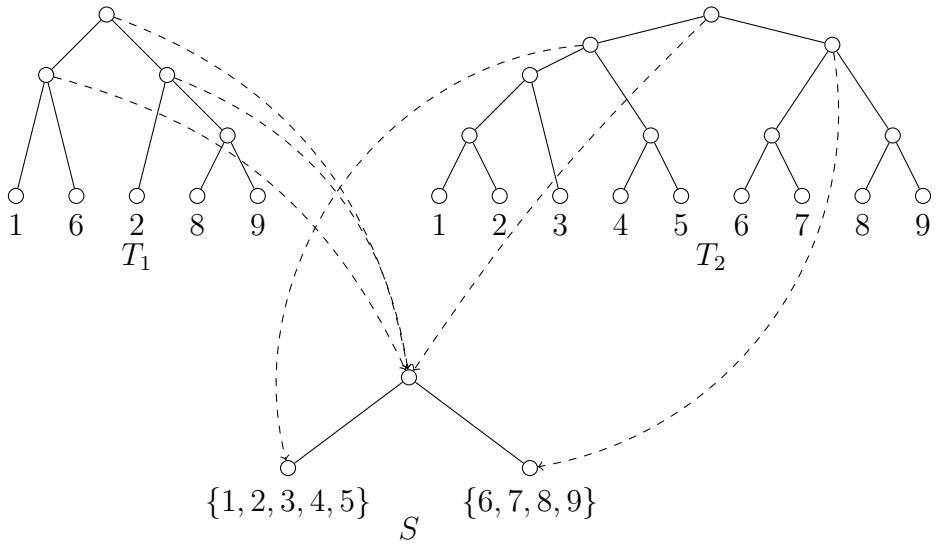

Figure 2: A set $\mathcal{F} = (T_1, T_2)$ of gene trees and the species tree S which is a bipartition $(\{1,2,3,4,5\},\{6,7,8,9\})$. For sake of clarity, we have labelled the leafs of T_1 and T_2 with the species of S. The only node with a pre-duplication is the root of T_1, since this node and its two children are mapped by $\mathcal{M}$ (*via* the dashed lines) to the root of S. Still for sake of clarity, the mapping is not drawn for all nodes of T_1 and T_2.

denote by $\Lambda(U)$ the set of its leaves. Given an internal node x of U, we denote by $U(x)$ the subtree of U rooted at node x, and by $\Lambda(U(x))$ the set of leaf labels of $U(x)$. When there is no ambiguity on the tree considered, we denote $\zeta(x) = \Lambda(U(x))$; $\zeta(x)$ is called the *cluster* of x. Given a tree U, we denote by $\texttt{lca}_T(u, v)$ the lowest common ancestor of two nodes u and v in a tree T.

Given a gene tree T and a species tree S, leaf-labelled by a set Λ, we define a mapping $\mathcal{M}$ (also called *least common ancestor mapping*) from the nodes of T to the nodes of S, defined as follows. $\mathcal{M}(x) = y$ where y is the node of S having minimum cluster such that $\zeta(x) \subseteq \zeta(y)$.

For example, in Fig. 1, according to $\mathcal{M}$, g_3 is mapped to s_1, since s_1 is the most recent common ancestor of s_4 and s_5 from which were sampled (represented as a function f) respectively the descendant g_4 and g_5 of g_3. Observe that, considering $\mathcal{M}$, any leaf of the gene tree is mapped to the unique leaf of S similarly labelled (according to Λ). Given $\mathcal{M}$, a node x in a gene tree T is a *duplication* if $\mathcal{M}(x) = \mathcal{M}(x')$, where x' is a child of x in T. The *duplication cost* $d(T, S)$ is defined as the number of duplications

induced in T. Given a forest $\mathcal{F} = \{T_1, T_2, \ldots, T_k\}$, the duplication cost of $\mathcal{F}$ with respect to S, denoted by $d(\mathcal{F}, S)$ is defined as $d(\mathcal{F}, S) = \sum_{i=1}^{k} d(T_i, S)$. The MINIMUM DUPLICATION problem [21, 12, 20, 22], is defined as follows:

MINIMUM DUPLICATION

- **Input**: A set $\mathcal{F}$ of gene trees.
- **Output**: A species tree S, with $\Lambda(S) \supseteq \bigcup_{T \in F} \Lambda(T)$.
- **Measure**: Duplication cost $d(\mathcal{F}, S)$.

A variant of the problem has been introduced in [28], where the goal is to compute the number of duplications induced by nodes of the gene trees mapped in the root of the species tree. Given a gene tree T and a species tree S, a node of T is a *pre-duplication* if x and one of its children are both mapped in the root of S. Formally, the MINIMUM DUPLICATION BIPARTITE problem is defined as follows:

MINIMUM DUPLICATION BIPARTITE

- **Input**: A set of labels Λ, a set $\mathcal{F}$ of gene trees on Λ.
- **Output**: A bipartition (Λ_1, Λ_2) of Λ.
- **Measure**: The number of pre-duplications.

2.1. Properties of the `lca` *mapping*

Let us introduce some fundamental properties that will be used in the rest of this paper. In the following, for ease, given a binary tree $T = (V, E)$ and a vertex $v \in V$, let us denote by v^L (resp. v^R) the left (resp. right) child of v, and by ζ_v the cluster of v *i.e.* the set of all leaves belonging to the subtree rooted in v. Moreover, for ease, ϑ_T will denote the root of the tree T.

Property 1. *Let T, T' be two gene trees labelled by the same sets of leaves Λ. Consider the bipartitions $b_1 = (\zeta(\vartheta_T^L), \zeta(\vartheta_T^R))$, $b_2 = (\zeta(\vartheta_{T'}^L), \zeta(\vartheta_{T'}^R))$ of Λ. Then either b_1 and b_2 are identical or any species tree S induces at least one duplication in the root of one of $\{T, T'\}$.*

Proof. The property follows easily from the observation that in any bipartition of the leaves Λ one of the set contains a leaf of $\zeta(\vartheta_T^L)$ and a leaf of $\zeta(\vartheta_T^R)$, or a leaf of $\zeta(\vartheta_{T'}^L)$ and a leaf of $\zeta(\vartheta_{T'}^R)$. □

Property 2. *Let $T = (V_T, E_T)$ be a gene tree and $S = (V_S, E_S)$ be a species tree. Let v be a vertex of V_T such that v has at least one child v', which is not a leaf. If there exists a vertex w of V_S such that (a) $\zeta(v) \setminus \zeta(v') \subseteq \zeta(w)$, (b) $\zeta(w) \not\supseteq \zeta(v)$ and (c) $\zeta(w) \cap \zeta(v') \neq \emptyset$, then v is duplicated.*

Proof. Let us consider the vertex w in S. Notice that, since (a) $\zeta(v) \setminus \zeta(v') \subseteq \zeta(w)$, (b) $\zeta(w) \not\supseteq \zeta(v)$ and (c) $\zeta(w) \cap \zeta(v') \neq \emptyset$, then there exists at least a label l, such that $l \in \zeta(v') \setminus \zeta(w)$ (otherwise $\zeta(w)$ would contain $\zeta(v)$). Furthermore, as $\zeta(w) \cap \zeta(v') \neq \emptyset$, it follows that v' and v are mapped to vertices of S that are on the path from w to ϑ_S. Let w' be the vertex of S where v' is mapped (*i.e.* $\mathcal{M}(v') = w'$). Note that w' is defined such that $\zeta(v') \subseteq \zeta(w')$ and $\nexists z$ such that $\zeta(v') \subseteq \zeta(z)$ and $|\zeta(z)| < |\zeta(w')|$. Since w' is an ancestor of w, it follows that $\zeta(v) \subseteq \zeta(w')$. Hence, v is mapped to w' (*i.e.* $\mathcal{M}(v) = w'$). As a consequence v is duplicated (see Fig. 3). □

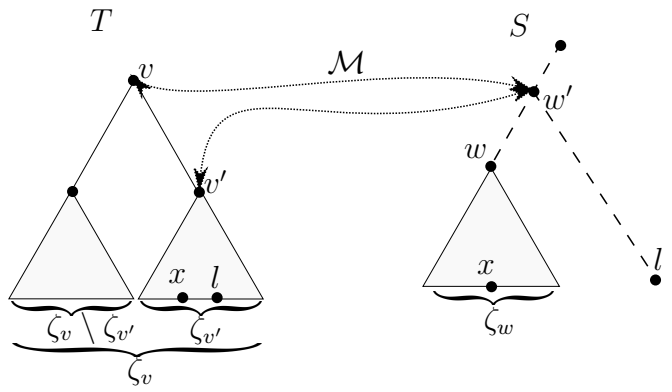

Figure 3: Illustration of Property 2 where (a) $\zeta(v) \setminus \zeta(v') \subseteq \zeta(w)$, (b) $\zeta(w) \not\supseteq \zeta(v)$ and (c) $\zeta(w) \cap \zeta(v') \neq \emptyset$.

3. On a tight inapproximability

We present a reduction from Minimum Vertex Cover on cubic graphs (MVCC) to the restriction of the Minimum Duplication problem – de-

noted MIN-5-DUP – where $\mathcal{F}$ consists of five gene trees, that is $\mathcal{F} = \{T_1, T_2, T_3, T_4, T_5\}$. The MVCC problem is defined as follows:

MINIMUM VERTEX COVER on cubic graphs (MVCC)

- **Input**: A cubic graph $G = (V_G, E_G)$ (*i.e.* every vertex has degree three).
- **Output**: A subset $V'_G \subseteq V_G$, such that for each edge $\{v_i, v_j\} \in E_G$, at least one of $\{v_i, v_j\}$ belongs to V'_G.
- **Measure**: $|V'_G|$.

In a first step (see Section 3.1), starting from a cubic graph $G = (V_G, E_G)$, we construct an associated input $\mathcal{F} = \{T_1, \ldots, T_5\}$ of MIN-5-DUP. Then in Section 3.2, we prove that (1) any species tree S such that $d(\mathcal{F}, S) < q = 6|V_G| + 3|E_G| + 1$ must induce duplications in the spine (the definition of spine is given afterwards) of trees $T_1, \ldots, T_4$, and (2) that our construction is indeed an L-reduction.

3.1. Extra definitions and construction

In order to define formally the gene trees, let us first define the central notion of *comb graph*. We will consider a specific subclass of comb graphs corresponding to a binary tree where all the internal nodes lie on a single simple (*i.e.* with no repeated vertices) path. We will nevertheless use the term comb graph in the following to denote those lasts. Given a sequence $L = \langle l_1, \ldots, l_k \rangle$ of k labels, let $C(L)$ denote the comb graph whose leaves are labelled according to a postorder traversal using L (*i.e.* $l_x \in L$, $1 \leq x \leq k-2$, is the label of the unique leaf of depth x, and l_{k-1}, l_k are both at level $k-1$). For example, in Fig. 1, the gene tree (a) corresponds to the comb graph $C(\langle g_2, g_4, g_5 \rangle)$.

Let us now define some operations on trees. Let $T_1 \triangle T_2$ be a tree obtained from two trees T_1 and T_2, by connecting the roots of T_1 and T_2 to a new vertex v which becomes the root of $T_1 \triangle T_2$. The insertion of T_2 in the edge e of T_1 denotes the operation that leads to a tree obtained from T_1 by replacing the edge $e = \{v, v'\}$ in T_1 by two edges $\{v, w\}$ and $\{w, v'\}$ and connecting the root of T_2 to the new vertex w. Given a binary tree $T = (V, E)$, with leaves labelled by Λ and $\Lambda' \subseteq \Lambda$, we define the *restriction of T to Λ'*, denoted

$T|\Lambda'$, as the subtree obtained from T by retaining only leaves with a label belonging to Λ' and by contracting all the internal vertices of degree 2.

We are now ready to define the gene trees $T_1, \ldots, T_5$. Roughly, we will associate with each vertex $v \in V_G$, a specific tree T_v and with each edge $e \in E_G$, two trees T_e^1, T_e^2. These trees will be then combined to build the gene trees $T_1, \ldots, T_5$. For ease, let us consider the following order on the edges of E_G, $\langle e_1, e_2, \ldots e_{|E_G|}\rangle$: $\forall e_x = \{v_i, v_j\}$, $e_y = \{v_h, v_k\}$ it holds $x < y$, where $i < j$ and $h < k$, either if $(i < h)$ or $(i = h$ and $j < k)$. Set $q = 6|V_G| + 3|E_G|$. According to this order, we define the following sequences of labels:

- $M_1 = \langle m_1^1, m_2^1, \ldots m_{|E_G|}^1\rangle$
- $M_2 = \langle m_1^2, m_2^2, \ldots m_{|E_G|}^2\rangle$
- $L = \langle l_1^1, l_1^2, l_2^1, l_2^2, \ldots l_{|E_G|}^1, l_{|E_G|}^2\rangle$
- $L_f = \langle f_{|V_G|+1}, f_{|V_G|}^1, \ldots, f_{|V_G|}^q, f_{|V_G|}, \ldots f_1^1, \ldots, f_1^q, f_1\rangle$

The sequences M_1, M_2, L encode the edges of the cubic graph G and belong to the subtrees (defined later) encoding the vertices and the edges of G. The labels are grouped as in the definition given above, since three labels m_x^1, m_y^1, m_z^1 belong to the same subtree of T_{v_i}, three labels m_x^2, m_y^2, m_z^2 belong to the same subtree of T_{v_i} and three labels of L belong to the same subtree of T_{v_i} (see Fig. 4).

The sequence L_f consists of labels associated with leaves connected to the spine of T_5 (a similar sequence is used for $T_1, \ldots, T_4$). This set is introduced to separate the subtrees encoding different vertices and edges of G (see Fig. 5).

Moreover, for ease of notation we denote the following subsequence of labels of L_f:

- $L_f^i = \langle f_i^1, \ldots, f_i^q\rangle$, with $1 \leq i \leq |V_G|$.

The subsequence L_f^i is used to define those labels of T_5 associated with a single index i, $1 \leq i \leq |V_G|$ (see Fig. 5).

Roughly, any edge e_x is represented by the four labels $\{m_x^1, m_x^2, l_x^1, l_x^2\}$. First of all, for any edge $e_x \in E_G$, let us build the two trees $T_{e_x}^1 = C(\langle l_x^1, m_x^2, l_x^2\rangle)$ and $T_{e_x}^2 = C(\langle l_x^2, m_x^2, l_x^1\rangle)$. Moreover, recall that G is cubic, therefore, any vertex has degree three. Then, for any $v_i \in V_G$ such that $e_x = \{v_i, v_j\}, e_y = \{v_i, v_{j'}\}, e_z = \{v_i, v_{j''}\} \in E_G$, with $j < j' < j''$, we build a tree

$$T_{v_i} = \left(C(\langle m_x^1, m_y^1, m_z^1\rangle) \triangle C(\langle l_x^k, l_y^{k'}, l_z^{k''}\rangle)\right) \triangle C(\langle m_x^2, m_y^2, m_z^2\rangle)$$

where k (resp. k', k'') is set to 1 if $i < j$ (resp. $i < j'$, $i < j''$); 2 otherwise (see Fig. 4).

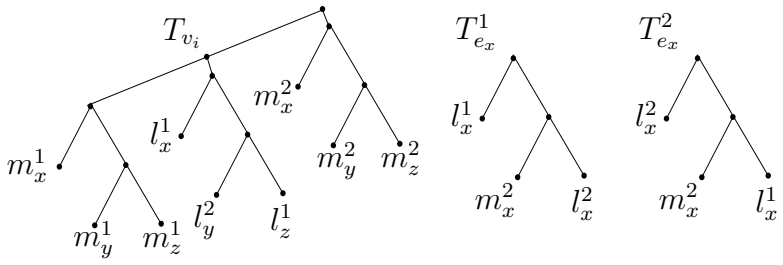

Figure 4: The trees T_{v_i}, $T^1_{e_x}$ and $T^2_{e_x}$ for $v_i \in V_G$ such that $e_x = \{v_i, v_j\}$, $e_y = \{v_i, v_{j'}\}$ and $e_z = \{v_i, v_{j''}\}$ and $i < j$, $i > j'$ and $i < j''$.

Now, we build the gene trees T_1 to T_5. We start from a comb graph where subtrees representing vertices and edges will be inserted in. Let T_5 be obtained from $C(L_f)$ by inserting the subtree $C(M_1) \triangle C(M_2)$ in the edge connecting f_1 and its parent (see Fig. 5).

Regarding the construction of T_1 to T_4, let us assume that we are also provided a 4-coloring $\lambda : V_G \mapsto \{1, 2, 3, 4\}$ of G (for example, by applying the polynomial-time greedy Welsh-Powell algorithm [29]). Let any T_i, $1 \leq i \leq 4$, be first defined as the following comb graph: $T_i = C(\langle f_1, f_2, \ldots, f_{|V_G|+1}, f^1_{|V_G|}, \ldots f^q_{|V_G|}, f^1_{|V_G|-1} \ldots, f^q_1 \rangle)$. We then insert, for each $v_i \in V_G$, the tree T_{v_i} in the edge connecting the parents of f_i and f_{i+1} in the gene tree T_x where $x = \lambda(v_i)$ (see Fig. 5). Moreover, for each $e_x = \{v_i, v_j\} \in E_G$ (ordered from e_1 to $e_{|E_G|}$), the tree $T^1_{e_x}$ is inserted in the edge connecting the parent of f_i and its other child in the gene tree T_x where $x = min\{1, 2, 3, 4\} \setminus \{\lambda(v_i), \lambda(v_j)\}$ (*i.e.* the gene tree having the smallest index and not containing either T_{v_i}, nor T_{v_j}). Finally, for each $e_x = \{v_i, v_j\} \in E_G$, the tree $T^2_{e_x}$ is inserted in the edge connecting the parent of f_i and its other child in the gene tree T_x where $x = max\{1, 2, 3, 4\} \setminus \{\lambda(v_i), \lambda(v_j)\}$ (*i.e.* the gene tree having the biggest index and not containing neither T_{v_i}, nor T_{v_j}). A sketch of this construction is given in Fig. 5.

Let P_x be the set of internal vertices in T_x, $1 \leq x \leq 4$, belonging to the path from the root of T_x to the parent $p^x_{|V_G|+1}$ of $f_{|V_G|+1}$. We define the *spine of any gene tree* T_x, $1 \leq x \leq 4$, as $P_x \setminus \{p^x_{|V_G|+1}\}$.

Recall that given a gene tree T and a species tree S, a vertex v of T is

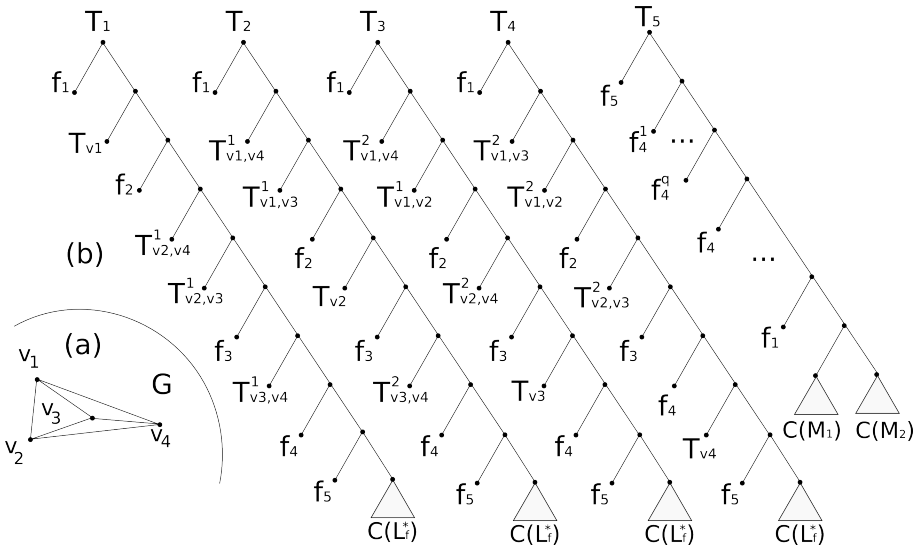

Figure 5: Gene trees T_1 to T_5 obtained from the cubic graph G where $L_f = \langle f_5, f_4^1, \ldots, f_4^q, \ldots, f_1^1, \ldots, f_1^q, f_1 \rangle$ and $\forall 1 \leq i \leq 4, \lambda(v_i) = i$. Each tree T_i, $1 \leq i \leq 4$, is obtained by inserting tree T_{v_j}, $T^x_{j,h}$ in the comb graph $C(\langle f_1, \ldots, f_5 \rangle)$. Notice that T_5 contains a comb graph $C(\langle f_5, \ldots, f_1 \rangle)$.

duplicated with respect to S if $\mathcal{M}(v) = \mathcal{M}(v')$ where v' is a descendant of v in T (*i.e.* v' belongs to T_v).

3.2. Correctness of the reduction

Before giving the details of the proofs, we will give an overview of the reduction. First of all, we will prove in Lemma 1, that all the gene trees in $\mathcal{F}$ are uniquely leaf-labelled. Then, we will prove (Lemma 2) that we can restrict ourselves to solutions, *i.e.* species trees, that contain an isomorphic copy of T_5, thus inducing duplications in each internal node of the spine of $T_1, \ldots T_4$. Then, applying the result of Lemma 3, we are able to relate the duplications of subtrees T_v, $v \in V_G$, with the corresponding vertices in the vertex cover of the graph G.

First of all, let us prove that, by construction, all the gene trees $T_1, \ldots, T_5$ are uniquely leaf-labelled.

Lemma 1. *The trees $T_1, \ldots, T_5$ are uniquely leaf-labelled trees.*

Proof. It is easy to see that T_5 is uniquely leaf-labelled by construction. Indeed, each of the three comb graphs $C(L_f)$, $C(M_1)$, $C(M_2)$ used to build

T_5 is uniquely leaf-labelled. Moreover, the trees $C(L_f)$, $C(M_1)$, $C(M_2)$ have pairwise disjoint sets of leaves.

Now, consider the gene trees T_1, T_2, T_3 and T_4. First, remark that each tree T_v, with $v \in V_G$, is uniquely leaf-labelled and so do the trees $T^1_{e_x}$ and $T^2_{e_x}$ (see Fig. 4). Then, consider the relative placement of T_v, $T^1_{e_x}$ and $T^2_{e_x}$ in the gene trees. More precisely, by construction, we have to ensure that a tree T_v, where $v \in V_G$ is incident to the three edges $e_x = \{v, v'\}$, $e_y = \{v, v''\}$ and $e_z = \{v, v'''\}$, does not belong to the same gene tree of $\{T_{v'}, T_{v''}, T_{v'''}\}$ and of $\{T^1_{e_x}, T^2_{e_x}, T^1_{e_y}, T^2_{e_y}, T^1_{e_z}, T^2_{e_z}\}$. This is indeed true since all those trees are associated with the gene trees considering their corresponding colors in the 4-coloring of G which ensures that (a) T_v, $T_{v'}$, $T_{v''}$ and $T_{v'''}$ belong to four different trees, (b) $\{T^1_{e_x}, T^2_{e_x}\}$ are inserted in the trees where $\{T_v, T_{v'}\}$ is not present (a similar property holds for $\{T^1_{e_y}, T^2_{e_y}\}$, and $\{T^1_{e_z}, T^2_{e_z}\}$). Moreover, the trees $T^1_{e_x}$ and $T^2_{e_x}$ do not belong to the same gene tree (which is the case by construction), and this concludes the proof. □

Let us now prove that we are interested only in solutions that induces a duplication in each node on the spine of T_x, $x \in \{1, \ldots, 4\}$.

First, let us consider the following order induced by the `lca` mapping $\mathcal{M}$. Consider three vertices v, v', v'' of a tree T and the following ordering of their lowest common ancestors: we write $\texttt{lca}_T(v, v') > \texttt{lca}_T(v', v'')$ ($\texttt{lca}_T(v, v') \geq \texttt{lca}_T(v', v'')$ respectively) when the depth of the `lca` of v', v'' is *greater* (*greater or equal* respectively) than the one of v, v'.

Lemma 2. *Let S be a solution of* Min-5-Dup *for the instance $\mathcal{F} = \{T_1, \ldots, T_5\}$. Then, either $d(\mathcal{F}, S) \geq 6|V_G| + 3|E_G| + 1$ or all the vertices on the spines of the gene trees T_1, T_2, T_3, T_4 are duplicated.*

Proof. Consider any species tree S and two leaves f_i, f_{i+1} of T_5, for a given $1 \leq i \leq n$. Let w^j_i (resp. w^j_{i+1}) be the parent of f_i (resp. f_{i+1}) in the gene tree T_j, with $j \in \{1, 2, 3, 4\}$. Let x_i (resp. x_{i+1}) denote the parent in T_5 of f_i (resp. f_{i+1}). Similarly, let x^z_i denote the parent in T_5 of f^z_i.

In what follows, we consider a label f^z_i, with $1 \leq z \leq q$, and we prove that, considering the previously mentioned mapping, either all the internal vertices on the path from x_i to x_{i+1} in T_5 are duplicated (hence $d(\mathcal{F}, S) \geq q = 6|V_G| + 3|E_G| + 1$) or all the internal vertices on the path from w^j_i (included) to w^j_{i+1} (not included) are duplicated in T_j, $1 \leq j \leq 4$. To do so, we will consider a case by case analysis based on the possible mappings of f_i, f_{i+1} and f^z_i in S.

More precesily, we have the following possible cases: either for each $f_i^z \in L_f^i$, $\texttt{lca}_S(f_i, f_i^z) \geq \texttt{lca}_S(f_{i+1}, f_i^z)$ (Case 1) or there exists a $f_i^z \in L_f^i$ such that $\texttt{lca}_S(f_i, f_i^z) < \texttt{lca}_S(f_{i+1}, f_i^z)$ (Case 2). For Case 1, we have two possible subcases: for each $f_i^z \in L_f^i$, $\texttt{lca}_S(f_i, f_i^z) \geq \texttt{lca}_S(f_{i+1}, f_i^z)$ (Case 1.a) or there exists a f_i^z such that $\texttt{lca}_S(f_i, f_i^z) \geq \texttt{lca}_S(f_{i+1}, f_i^z)$ (Case 1.b).

Consider the subtree of $S' = S|\,(\{f_i\} \cup \{f_{i+1}\} \cup (\bigcup_z \{f_i^z\}))$. Intuitively, there exists a vertex x of S' such that $\zeta(x) = (\{f_i\} \cup (\bigcup_z \{f_i^z\}))$, hence $f_{i+1} \notin \zeta(x)$ (Case 2) or not (Case 1). Case 1 can have two possible subcases: there exists a vertex y of S' such that $\zeta(y) = (\{f_{i+1}\} \cup (\bigcup_z \{f_i^z\}))$ (Case 1.a) or not (Case 1.b), in which case there is a vertex w of S' such that $f_i, f_{i+i} \in \zeta(w)$, and $f_i^z \notin \zeta(w)$, for some z.

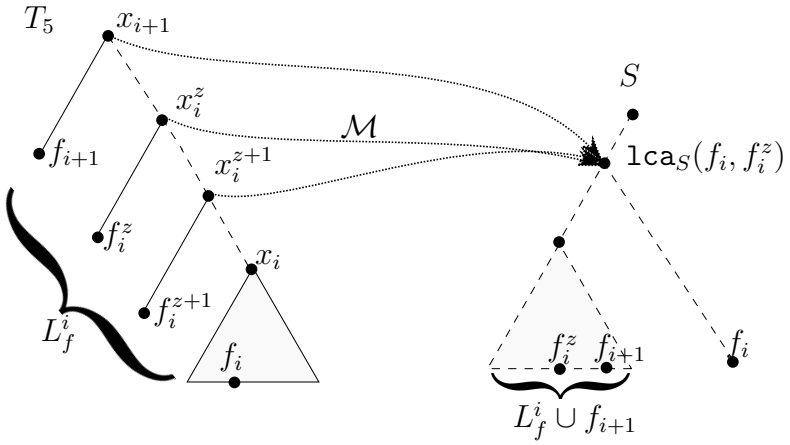

Figure 6: Illustration of Lemma 2, when for each $f_i^z \in L_f^i$, $\texttt{lca}_S(f_i, f_i^z) > \texttt{lca}_S(f_{i+1}, f_i^z)$. Vertices $x_{i+1}, \ldots, x_i^{z+1}$ of T_5 are duplicated.

(**Case 1**) Assume that for each $f_i^z \in L_f^i$, $\texttt{lca}_S(f_i, f_i^z) \geq \texttt{lca}_S(f_{i+1}, f_i^z)$. Notice that there exists two possible cases: $\texttt{lca}_S(f_i, f_i^z) > \texttt{lca}_S(f_{i+1}, f_i^z)$, for each f_i^z (Case 1.a, see Fig. 6), or there exists at least one f_i^z such that $\texttt{lca}_S(f_i, f_i^z) > \texttt{lca}_S(f_{i+1}, f_i)$ (Case 1.b, see Fig. 7).

(**Case 1.a**) In this case $\texttt{lca}_S(f_i, f_i^z) > \texttt{lca}_S(f_{i+1}, f_i^z)$, for each f_i^z. Property 2 applies to each internal vertex between x_i, x_{i+1} excluding x_i with $v = x_i^z$, $v' = x_i^{z+1}$, $x = f_i^{z+1}$ and $l = f_i$ (see Fig. 6). Hence $d(\mathcal{F}, S) \geq q = 6|V_G| + 3|E_G| + 1$ since each x_i^j with $1 \leq j \leq q-1$ and x_{i+1} are duplicated.

(**Case 1.b**) In this case for some f_z^i, it holds that $\texttt{lca}_S(f_i, f_i^z) > \texttt{lca}_S(f_{i+1}, f_i)$. Let us consider the leaves in $\zeta(w_i^j) \setminus \zeta(w_{i+1}^j)$ and let S_i^j be

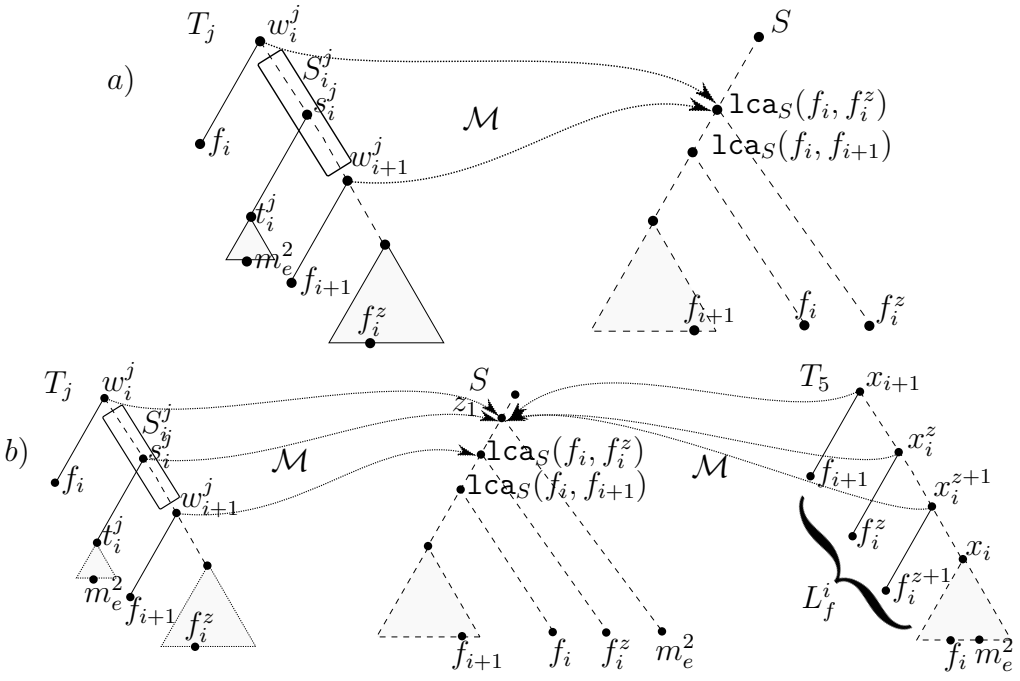

Figure 7: Illustration of Lemma 2 when there exists at least one f_i^z such that $\texttt{lca}_S(f_i, f_i^z) > \texttt{lca}_S(f_{i+1}, f_i)$. In case a) the vertices of T_j, $1 \leq j \leq 4$, between w_i^j and w_{i+1^j} are duplicated. In case b), since vertex w_{i+1}^j is not mapped in z_1, then all the vertices between x_{i+1} and x_i^{z+1} in T_5 are duplicated.

the set of internal vertices of T_j, $1 \leq j \leq 4$, between w_{i+1}^j and w_i^j. Property 2 applies to the vertex w_i^j, as $\texttt{lca}_S(f_{i+1}, f_i)$ contains f_i, f_i^z but not f_{i+1}, which is contained in both $\zeta(w_{i+1}^j)$ and $\zeta(w_i^j)$. Hence w_i^j is duplicated (see Fig. 7a). Now, we have to consider the vertices between w_i^j and w_{i+1}^j.

Consider the lowest vertex $s_i^j \in S_i^j$ not duplicated and denote by t_i^j its child which is not on the spine of T_j, $1 \leq j \leq 4$. Let z_1 be the vertex of S where s_i^j is mapped (*i.e.* $\mathcal{M}(s_i^j) = z_1$), and notice that $z_1 \geq \texttt{lca}_S(f_i^z, f_i)$. Since s_i^j is not duplicated, then the cluster of one of the children of z_1 contains $\zeta(t_i^j)$, while the other contains $\{f_i^z, f_i, f_{i+1}\}$, for each $1 \leq z \leq q$. But then, since there exists a $m_e^2 \in \zeta(s_i^j)$, it follows that Property 2 applies to each internal vertex between x_i, x_{i+1} excluding x_i with $v = x_i^z$, $v' = x_i^{z+1}$, $x = f_i^{z+1}$ and $l = m_e^2 i$, for all $1 \leq z \leq q$ (see Fig. 7b). Hence $d(\mathcal{F}, S) \geq q = 6|V_G| + 3|E_G| + 1$ since each x_i^j with $1 \leq j \leq q-1$ and x_{i+1} are duplicated.

(**Case 2**) Now, let us consider the case $\texttt{lca}_S(f_i, f_i^z) < \texttt{lca}_S(f_{i+1}, f_i^z)$.

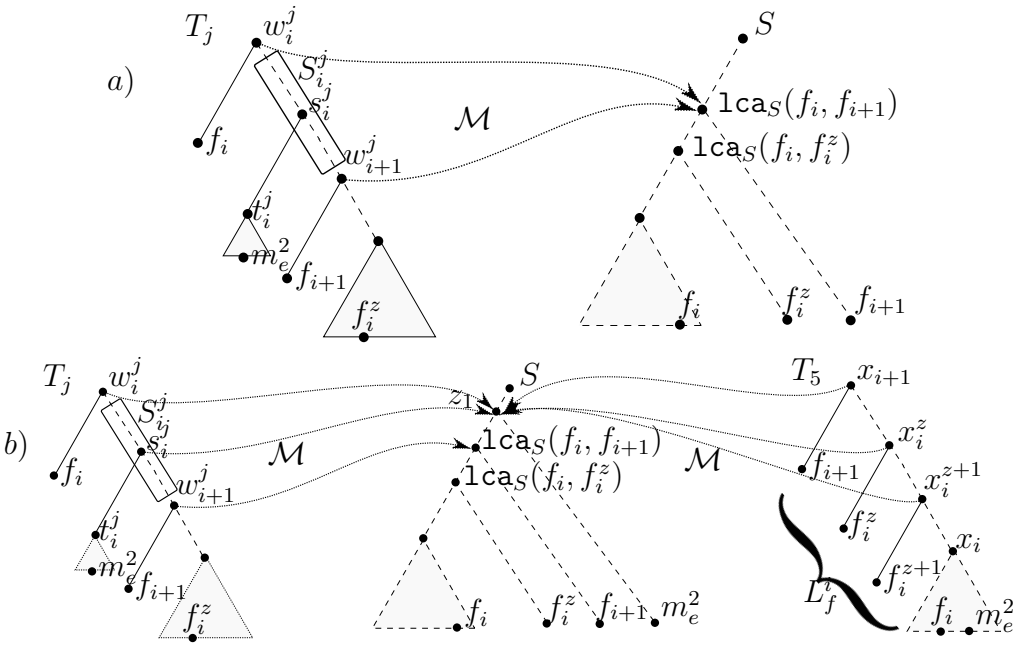

Figure 8: Illustration of Lemma 2 when $\texttt{lca}_S(f_i, f_i^z) < \texttt{lca}_S(f_{i+1}, f_i^z)$. In case a) the vertices of T_j, $1 \leq j \leq 4$, between w_i^j and w_{i+1^j} are duplicated. In case b), due to the leaf label m_e^2 in S, vertex w_{i+1}^j is not mapped in z_1, and all the vertices between x_{i+1} and x_i^{z+1} in T_5 are duplicated.

Property 2 applies to the vertex w_i^j, as $\zeta(w_i^j) = \zeta(s_i^j) \backslash \{f_i\}$, while $\texttt{lca}_T(f_i, f_i^z)$ contains f_i, but not f_{i+1}. Hence w_i^j is duplicated (see Fig. 8a). Now, we have to consider the vertices between w_i^j and w_{i+1}^j.

Consider the lowest vertex $s_i^j \in S_i^j$ not duplicated and denote by t_i^j its child which is not on the spine of T_j, $1 \leq j \leq 4$. Let z_1 be the vertex of S where s_i^j is mapped (*i.e.* $\mathcal{M}(s_i^j) = z_1$), and notice that $z_1 \geq \texttt{lca}_S(f_i^z, f_{i+1})$. Since s_i^j is not duplicated, then the cluster of one of the children of z_1 contains $\zeta(t_i^j)$, while the other contains $\{f_i^z, f_i, f_{i+1}\}$, for each $1 \leq z \leq q$. But then, since there exists a $m_e^2 \in \zeta(s_i^j)$, it follows that Property 2 applies to each internal vertex between x_i, x_{i+1} (excluding x_i), with $v = x_i^z$, $v' = x_i^{z+1}$, $x = f_i^{z+1}$ and $l = m_e^2$, for all $1 \leq z \leq q$ (see Fig. 8b). Hence $d(\mathcal{F}, S) \geq q = 6|V_G| + 3|E_G| + 1$ since each x_i^j with $1 \leq j \leq q-1$ and x_{i+1} are duplicated.

Since we have shown that for each pair of leaves f_i, f_{i+1}, either $d(\mathcal{F}, S) \geq 6|V_G| + 3|E_G| + 1$, or all the internal nodes between w_i^x (included) and w_{i+1}^x (not included) are duplicated, it follows that we have proved the lemma. □

While in the previous lemma we have focused on the duplications induced by vertices of the spine of the gene trees $T_1, \ldots, T_4$, in what follows, we will focus on the duplications induced in the subtrees representing the vertices and edges (*i.e.* T_v, T_e^1, T_e^2).

Lemma 3. *Let S be a solution of* Min-5-Dup *over instance $\mathcal{F} = \{T_1, \ldots, T_5\}$ and let $T^1_{e_x}$, $T^2_{e_x}$, T_{v_i}, T_{v_j} be four subtrees of $T_1, \ldots, T_4$, s.t. $e_x = \{v_i, v_j\} \in E_G$ and $i < j$. Then (1) the root of at least one of $T^1_{e_x}$, $T^2_{e_x}$ is duplicated with respect to S; (2) the roots of at least two of $T^1_{e_x}$, $T^2_{e_x}$, T_{v_i}, T_{v_j} are duplicated with respect to S.*

Proof. (1) The proof follows from Property 1, since the roots of $T^1_{e_x}$, $T^2_{e_x}$ induces two different bipartitions of the sets $\{m^2_x, l^1_x, l^2_x\}$.

(2) Now, let us prove the second part of the lemma. We have shown that any species tree S induces a duplication in the root of at least one of $T^1_{e_x}$, $T^2_{e_x}$. If S induces a duplication in the roots of both $T^1_{e_x}$ and $T^2_{e_x}$, then the lemma holds. Hence assume that S induces a duplication in exactly one of $T^1_{e_x}$, $T^2_{e_x}$, w.l.o.g. $T^1_{e_x}$. Thus, assume that S does not induce a duplication in the root of $T^2_{e_x}$.

Let us define $L_x = \{m^1_x, m^2_x, l^1_x, l^2_x\}$ and consider $T_{v_i}|L_x$, $T_{v_j}|L_x$. The roots of $T_{v_i}|L_x$ and $T_{v_j}|L_x$ induce, by construction, the following bipartitions $B(v_i) = (\{m^1_x, l^1_x\}; \{m^2_x\})$ and $B(v_j) = (\{m^1_x, l^2_x\}; \{m^2_x\})$.

Let v be the vertex of S, which is the lowest common ancestor of $\{m^2_x, l^1_x, l^2_x\}$. Since we have assumed that the root of $T^2_{e_x}$ is not duplicated, it follows that the subtree rooted at v restricted to $\{m^2_x, l^1_x, l^2_x\}$ must induce the bipartition $(\{m^2_x, l^1_x\}; \{l^2_x\})$ (as defined in $T^2_{e_x}$). Now, assume that both the root of T_{v_i} and the root of T_{v_j} are mapped to v and consider where the leaf m^1_x is possibly placed in the subtree $S(v)$. If m^1_x is in the same set of the bipartition with l^2_x, then the root of T_{v_i} is duplicated (as $B(v_i) = (\{m^1_x, l^1_x\}; \{m^2_x\})$, see Fig. 9a). If m^1_x is in the same set of the bipartition with l^1_x and m^2_x, then the root of T_{v_j} is duplicated, (as $B(v_j) = (\{m^1_x, l^2_x\}; \{m^2_x\})$, see Fig. 9b).

Assume now that the root of T_{v_i} or the root of T_{v_j}, w.l.o.g. $\vartheta_{T_{v_i}}$, is not mapped to v. Then, $\vartheta_{T_{v_i}}$ is either mapped to a descendant v' of v on the path from v to $\texttt{lca}_S(m^2_x, l^1_x)$ or to an ancestor v'' of v. If $\vartheta_{T_{v_i}}$ is mapped to v' then m^1_x is also a descendant of v' leading to a duplication of the root of T_{v_j} (see Fig. 10a). If $\vartheta_{T_{v_i}}$ is mapped to v'', then v'' induces the bipartion $(\{l^2_x, m^2_x, l^1_x\} \cup X; \{m^1_x\} \cup Y)$, for some sets $X, Y \subset \Lambda$, leading to a duplication of the root of T_{v_j} (see Fig. 10b). A similar proof can be derived if $\vartheta_{T_{v_j}}$, is not mapped to v.

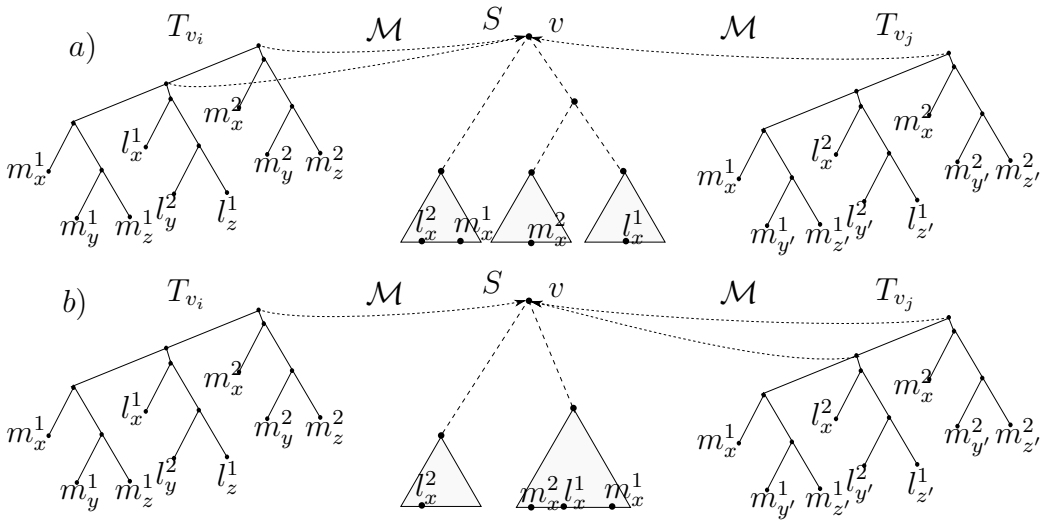

Figure 9: Illustration of the first part of Lemma 3. Case a) shows that, when the leaf labeled by $m_{1,x}$ is placed in the same set of the bipartition with l_x^2, the root of T_{v_i} is duplicated. Case b) shows that, when the leaf labeled by $m_{1,x}$ is placed in the same set of the bipartition with l_x^1, the root of T_{v_j} is duplicated.

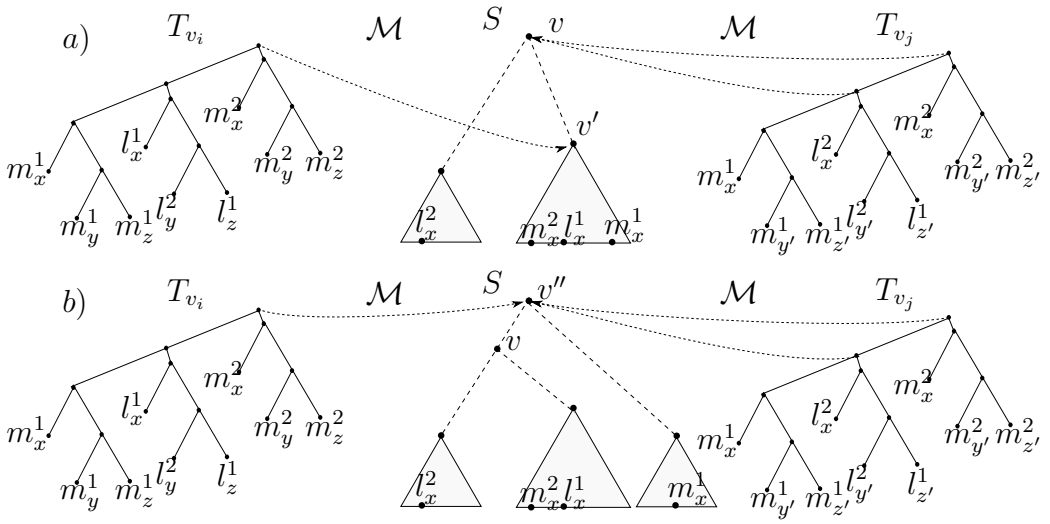

Figure 10: Illustration of the second part of Lemma 3. Case a) shows that, when the root of T_{v_i} is mapped to a descendant v' of v, the root of T_{v_j} is duplicated. Case b) shows that, when the root of T_{v_i} is mapped to an ancestor v'' of v, the root of T_{v_j} is duplicated.

□

Applying Lemma 2 and Lemma 3, we can prove the following fundamental result.

Lemma 4. *Let $G = (V_G, E_G)$ be an instance of* MVCC *and let $\mathcal{F} = \{T_1, \ldots, T_5\}$ be the corresponding instance of* Min-5-Dup*. Then, starting from a cover V'_G of G, we can compute in polynomial time a solution S of* Min-5-Dup *for $\mathcal{F} = \{T_1, \ldots, T_5\}$ s.t. $d(\mathcal{F}, S) \leq 5|V_G| + 3|E_G| + |V'_G|$; starting from a solution S of* Min-5-Dup *for $\mathcal{F}$ s.t. $d(\mathcal{F}, S) \leq 5|V_G| + 3|E_G| + p$, we can compute in polynomial time a cover of G of size at most p.*

Proof. First, consider a cover V'_G of $G = (V_G, E_G)$, we define a solution S to Min-5-Dup of cost at most $5|V_G| + 3|E_G| + |V'_G|$ (see Fig. 11 for an example). Define first S' as a tree isomorphic to T_5. S is obtained by inserting some subtrees in S'. More precisely, consider the subtree of S' having as leaf set $M_1 \cup M_2$. Let x be the root of this subtree, with children x^l, x^r Define the following comb graphs K_1 and K_2 (the order of the leaves in the two comb graphs is induced by the order on the corresponding edges of the graph G, and if two leaves l^1_x, l^2_x belong to the same comb graph, then $l^1_x < l^2_x$). Let $e_x = \{v_i, v_j\}$, $e_y = \{v_i, v_h\}$, $e_z = \{v_i, v_k\}$ be the three edge incident on v_i. Assume that v_i is the a-th (b-th, c-th respectively) vertex in $\{v_i, v_j\}$ ($\{v_i, v_h\}$, $\{v_i, v_k\}$ respectively), where $a, b, c \in \{1, 2\}$. This means that by construction the subtree $T(v_i)$ contains the leaves l^a_x, l^b_y, l^c_z. Then K_1 is a comb graph on the set L_1 defined as follows:

$$L_1 = \{l^a_x, l^b_y, l^c_z : v_i \in (V_G \setminus V'_G)\}$$

The comb graph K_2 is on the set $L_2 = L \setminus L_1$.

The two comb graphs K_1 and K_2 are inserted in the edges $\{x, x^l\}$ and $\{x, x^r\}$ (respectively). Next, we will show the duplication induced by S into the subtrees $T^1_{e_x}$, $T^2_{e_x}$ and T_{v_i}.

First, assume that $v_i \in (V_G \setminus V'_G)$. Then the corresponding subtree $T(v_i)$ does not contain duplications as it is isomorphic to $S|\Lambda(T(v_i))$.

Assume that $v_i \in V'_G$. Then by construction the subtree $T(v_i)|\{m^1_x, m^1_y, m^1_z\}$ ($T(v_i)|\{m^2_x, m^2_y, m^2_z\}$, $T(v_i)|\{l^a_x, l^b_y, l^c_z\}$ respectively) is isomorphic to the subtree $S|\{m^1_x, m^1_y, m^1_z\}$ ($S|\{m^2_x, m^2_y, m^2_z\}$, $S|\{l^a_x, l^b_y, l^c_z\}$ respectively). A duplication is induced in the root of $T(v_i)$ as $l^a_x, l^b_y, l^c_z \in L_2$ (hence in $\Lambda(K_2)$) while $m^1_x, m^1_y, m^1_z \in L_1$ (hence in $\Lambda(K_1)$).

Now, consider the subtrees $T^1_{e_x}$, $T^2_{e_x}$ associated with the edge $e_x = \{v_i, v_j\}$. If both $v_i, v_j \in V'_G$, then all the leaves l^1_x, l^2_x, m^2_x belongs to L_2 (hence to $\Lambda(K_2)$). Since we assume that $l^1_x < l^2_x$ in the order of leaves of K_2, it follows that no duplication is induced in $T^1_{e_x}$, and a duplication is induced in the root of $T^2_{e_x}$. Assume that exactly one of v_i, v_j belongs to V'_G (w.l.o.g. $v_i \in V'_G$). Then the leaves l^1_x, m^2_x belongs to L_2 (hence to $\Lambda(K_2)$), while $l^2_x \in L_1$. It follows that no duplication is induced in $T^1_{e_x}$, and a duplication is induced in the root of $T^2_{e_x}$.

Hence duplications are induced in: (1) the root of exactly one of the subtrees $T^1_{e_x}$, $T^2_{e_x}$; (2) the root of each subtree T_{v_i}, where $v_i \in V'_G$. Since all the nodes on the spine of each T_x are duplicated, it follows that S induces $5|V_G| + 3|E_G| + |V'_G|$ duplications.

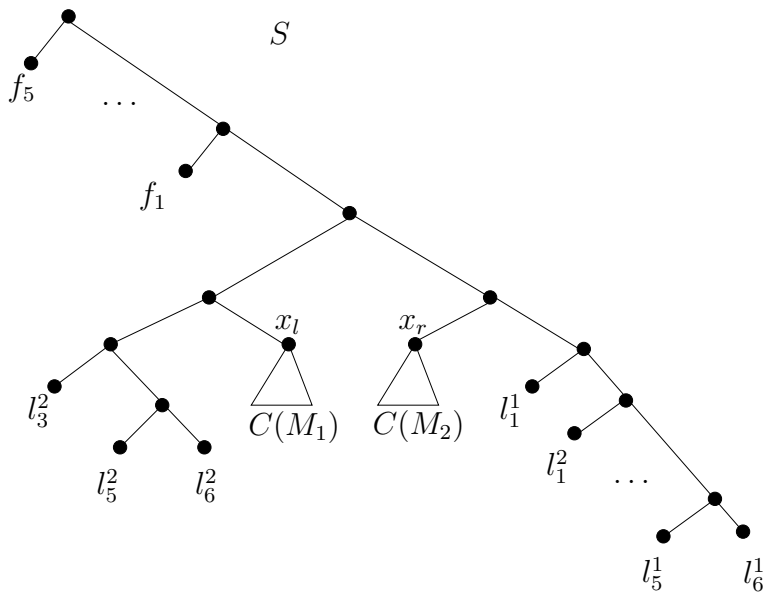

Figure 11: A solution S for the instance of Fig. 5, where the vertex cover $V'_G = \{v_1, v_2, v_3\}$. The labels l^2_3, l^2_5, l^2_6 belongs to the tree $T(v_4)$ (and notice that v_4 is the only vertex not in V'_G).

Now, consider a species tree S inducing at most $5|V_G| + 3|E_G| + p$ duplications. By Lemma 2, we can assume that S induces a duplication in the spine of each tree T_x, with $x \in \{1, \ldots, 4\}$. Now, we compute a vertex cover V'_G of G of size p as follows. For each subtree T_{v_i} such that a duplication is induced in the root of $T(v_i)$, add the corresponding vertex v_i to V'_G. If,

for some edge $\{v_i, v_j\}$, a duplication is not induced in the root of subtrees T_{v_i}, T_{v_j}, then add one of v_i, v_j to V'_G. By construction and by Lemma 3, for each edge $\{v_i, v_j\} \in E_G$, at least one of v_i, v_j belongs to V'_G. Since all the nodes on the spine of each T_x is duplicated (hence a total of $5|V_G| + 2|E_G|$ duplications), it follows that $|V'_G| = p' \leq p$, hence the lemma holds. □

Lemma 4 concludes the reduction.

Theorem 1. *The* Minimum Duplication *problem is* APX*-hard, even when the input consists of five uniquely leaf-labelled gene trees.*

Proof. First, notice that in a cubic graph $G = (V_G, E_G)$, $|E_G| = \frac{3}{2}|V_G|$, and a vertex cover V'_G of G has size at least $\frac{|V_G|}{4}$. Hence by Lemma 4 it follows that we have designed an L-reduction from MVCC to Min-5-Dup. Since MVCC is APX-hard [30], provided our L-reduction, we can conclude that Min-5-Dup is APX-hard. □

4. A randomized approach

In this section, we investigate the complexity of the Minimum Duplication Bipartite problem and show that it can be solved efficiently by a randomized algorithm when the input gene trees have bounded depth. A randomized algorithm can be seen simply as an algorithm that is allowed to do some random decisions as it processes the input. Whereas defining a randomized algorithm is quite easy, analyzing its performance is more complicated. Indeed, first, one has to compute the probability of success of the randomized algorithm (*i.e.* probability to end up with an optimal solution). Then, one can amplify the probability of success simply by repeatedly running the algorithm, with independent random choices, and taking the best solution founded. If one, moreover, prove that the overall running time required to get a high probability of success is polynomial in the size of the input, then it implies that the problem is randomized polynomial (in RP-class). For further details on randomized algorithms, the reader should consider the book of Kleinberg and Tardos [31].

In order to prove that the Minimum Duplication Bipartite problem is randomized polynomial, we first provide a randomized algorithm for a variant of the Minimum Cut problem, called Minimum Cut in Colored Graph. Then, we will prove that the Minimum Duplication Bipartite

problem can be translated into a Minimum Cut in Colored Hypergraph problem that can be solved efficiently applying our randomized algorithm on hypergraphs with bounded hyperedges degree. It is of importance to note that, as far as we know, this is the first attempt of solving by randomization the minimum cut in colored hypergraph. Providing a randomized algorithm for general hypergaphs with unbounded hyperedges degree is still open.

Let us first introduce the Minimum Cut in Colored Graph problem:

Minimum Cut in Colored Graph

• **Input**: A set of colors $\mathcal{C}$, and an undirected colored graph $G = (V, E)$ where any edge is colored with a color from $\mathcal{C}$.

• **Output**: A colored cut of G – that is a partition of V into two non-empty sets A and B.

• **Measure**: The number of colors used by the edges having one end in A and the other in B.

For ease, let $\texttt{col} : E \mapsto \mathcal{C}$ be a function returning the color of a given edge and $\texttt{mul}(c) = |\{e : e \in E \text{ and } \texttt{col}(e) = c\}|$ be a function returning the multiplicity of a given color. Moreover, for sake, given a graph $G = (V, E)$, let $\texttt{col}(G) = \bigcup_{e \in E} \texttt{col}(e)$ denote the set of colors used in G. Let us now describe an algorithm inspired by the folklore Contraction Algorithm [31] used for solving the classical Minimum Cut problem (*i.e.* minimizing the number of edges having one end in A and the other in B) on uncolored graph by randomized algorithm.

As in [31], our Colored Contraction Algorithm uses a connected multigraph $G = (V, E)$ – that is an undirected graph that is allowed to have more than one edge between the same pair of vertices – which is moreover colored. The algorithm starts by choosing, uniformly at random, a color $c \in \texttt{col}(G)$ and contracting any edge $e \in E$ such that $\texttt{col}(e) = c$ (and thus all such edges). Contracting an edge $\{u, v\} \in E$ will produce a new graph $G' = (V', E')$ in which u and v are identified as a single new vertex w whereas all other vertices are keeping their original identity (*i.e.* $V' = \{V \cup \{w\}\} \setminus \{u, v\}$). In G', $E' = \{E \cup \{\{w, v''\} : v' \in \{u, v\}, \{v', v''\} \in E\}\} \setminus \{\{v', v''\} : v' \in \{u, v\}, v'' \in V\}$. Roughly, E' is a copy of E where any edge $\{u, v\}$ has been removed whereas any other edge has been preserved, but if one of its ends was equal to u or v, then this end is updated to be

equal to the new node w. Note that the contraction operation may end up in a multigraph even when starting from a classical graph G. In this process, contracting all the edges that have the selected color c roughly corresponds to a sequence of $\texttt{mul}(c)$ contractions, each reducing the number of vertices by one. Colored Contraction Algorithm then continues recursively on G', by choosing, uniformly at random, a color $c \in \texttt{col}(G')$ and contracting any edge $e \in E$ such that $\texttt{col}(e) = c$. As these recursive calls proceed, the vertices of V' should be viewed as *supervertices*: each supervertex w corresponds to the subset $\mathcal{S}(w) \subseteq V$ that has been "swallowed up" in the contractions that produced w. The algorithm ends when it reaches a graph G' with only two super-vertices v_A and v_B. We output $(A = \mathcal{S}(v_A), B = \mathcal{S}(v_B))$ as the colored-cut found by the algorithm.

Let us now analyze the performance of the Colored Contraction Algorithm – which cannot be derived directly from the one of the original Contraction Algorithm. Since the algorithm is making random choices, there is some probability that it will succeed in finding a minimum colored-cut (and some probability that it would not). In order to prove that this algorithm is worthwhile, we will prove that the probability of success is only polynomially small; inducing that, by running the algorithm a polynomial number of times and returning the best colored-cut found in any run, one would be able to produce an optimal colored-cut with high probability.

Theorem 2. *The* Colored Contraction Algorithm *returns an optimal colored-cut G with probability at least $(|V|^{2k})^{-1}$ where $k = \text{MAX}_{c \in \mathcal{C}} \texttt{mul}(c)$*

Proof. Let us assume that the optimal minimum colored-cut (A, B) of G is of size OPT; that is the set of edges having one end in A and the other end in B (referred afterwards as the *cut-set*) is colored using OPT colors of $\mathcal{C}$. Note that unlike the classical minimum cut problem, the goal here is to minimize the number of colors in the cut-set itself. Moreover, let $G_{\text{OPT}} = G[A \cup B, \{(u, v) : (u, v) \in E \text{ and } u \in A, v \in B\}]$ corresponds to the bipartite graph representing the cut-set of (A, B). In order to compute a lower bound on the probability that the Colored Contraction Algorithm returns the minimum colored-cut (A, B), we first notice some important properties.

First, remark that any vertex $v \in V$ cannot have a degree less than OPT. Indeed, otherwise, $(\{v\}, V \setminus \{v\})$ would correspond to a colored-cut inducing at most OPT-1 colors, contradicting our hypothesis that (A, B) is an optimal minimum colored-cut of G. Therefore, any vertex of G is of degree at least OPT; inducing the following lower bound on E: $|E| \geq \frac{\text{OPT}|V|}{2}$. We know

moreover that, since each color of $\mathcal{C}$ can be used at most $k = \text{MAX}_{c\in\mathcal{C}}\texttt{mul}(c)$ times in E, we have that $|E| \leq k\cdot|\mathcal{C}|$. This leads to the following inequalities.

$$|V| \cdot \text{OPT} \leq 2 \cdot |E| \leq 2k \cdot |\mathcal{C}| \tag{1}$$

Let us now evaluate the probability $Pr[F_j]$ that the COLORED CONTRACTION ALGORITHM fails at the j^{th} step of the recursion (that is when already $j-1$ contractions have been done). Considering what could go wrong in the j^{th} step of the COLORED CONTRACTION ALGORITHM, one can check that the unique issue would be that the uniformly at random choice of a color c unfortunately select one color of the set of OPT colors used by the cut-set – which will be then contracted inducing that the algorithm would not be able to find the optimal colored-cut (A, B) since at least a node of A and a node of B would be both contracted into the same supervertex. Hence the probability that an edge of the current graph G' is both in the optimal cut-set and contracted is at most $\frac{\text{OPT}}{|\mathcal{C}'|}$, since there are at most OPT edges to be chosen among $|\mathcal{C}'|$ edges, where $\mathcal{C}' = \texttt{col}(G')$. According to Inequality 1, considering that the graph at j^{th} step is G' and $\mathcal{C}' = \texttt{col}(G')$

$$Pr[F_j] \leq \frac{\text{OPT}}{|\mathcal{C}'|} \leq \frac{2k \cdot |\mathcal{C}'|}{|V'| \cdot |\mathcal{C}'|} = \frac{2k}{|V'|} \tag{2}$$

The colored-cut (A, B) will actually be returned by the algorithm if no edge of the cut-set is contracted in any of the at most $|V| - 2$ iterations. If we write S_j for the event that an edge of the cut-set has not been contracted until the j^{th} step, then, according to Inequality 2, $Pr[S_j] \geq 1 - Pr[F_j] = 1 - \frac{2k}{|V'|}$ where the graph at j^{th} step is $G' = (V', E')$. For ease, let us consider the sequence of color choices as being $\mathcal{S}_c = (c_1, c_2 \ldots)$ and $\lambda_j = \sum_{i<j \text{ and } c_i\in\mathcal{S}_c} \texttt{mul}(c_i)$. On the whole the probability that the COLORED CONTRACTION ALGORITHM returns the optimal colored-cut (A, B) is thus at least

$$Pr[Success] \geq \prod_{i=0}^{\lambda_1-1}(1-\frac{2k}{|V|-i})\cdot\prod_{i=\lambda_2}^{\lambda_3-1}(1-\frac{2k}{|V|-i})\ldots\prod_{i=\lambda_{|\mathcal{S}_c|-1}}^{\lambda_{|\mathcal{S}_c|}-1}(1-\frac{2k}{|V|-i}) \tag{3}$$

$$\geq \prod_{i=0}^{\lambda_1-1}(\frac{|V|-i-2k}{|V|-i})\cdot\prod_{i=\lambda_2}^{\lambda_3-1}(\frac{|V|-i-2k}{|V|-i})\ldots\prod_{i=\lambda_{|\mathcal{S}_c|-1}}^{\lambda_{|\mathcal{S}_c|}-1}(\frac{|V|-i-2k}{|V|-i}) \tag{4}$$

$$\geq \prod_{i=0}^{\lambda_{|\mathcal{S}_c|}-1} \left(\frac{|V|-i-2k}{|V|-i}\right) = \frac{\cancel{|V|-2k}}{|V|} \cdots \frac{|V|-2k-2k}{\cancel{|V|-2k}} \cdots \frac{|V|-(\lambda_{|\mathcal{S}_c|}-1)-2k}{\cancel{|V|-(\lambda_{|\mathcal{S}_c|}-1)}} \tag{5}$$

$$\geq \frac{\prod_{i=2k}^{\lambda_{|\mathcal{S}_c|}-1} |V|-i-2k}{\prod_{i=0}^{2k-1} |V|-i} \geq \frac{1}{|V|^{2k}} = (|V|^{2k})^{-1} \tag{6}$$

□

Then according to Theorem 2, we know that a single run of the Colored Contraction Algorithm fails to find an optimal colored-cut with probability at most $(1-(|V|^{2k})^{-1})$. One can then amplify the probability of success simply by repeatedly running the algorithm, with independent random choices, and taking the best colored-cut found. It is known that the function $(1-n^{-1})^n$ converges monotonically from $\frac{1}{4}$ up to $\frac{1}{e}$ as n increases from 2 [31]. Thus, if we run the algorithm $|V|^{2k}$ times, then the probability that we fail to find an optimal colored-cut in any run is at most $(1-(|V|^{2k})^{-1})^{|V|^{2k}} \leq \frac{1}{e}$. As usually done, it is easy to even reduce more the failure probability with further repetitions by running the algorithm $|V|^{2k} \ln |V|$ times which induces a probability of failure of at most $e^{-\ln |V|} = \frac{1}{|V|}$. Overall, the running time required to get a high probability of success is polynomial in $|V|$ if k is bounded, since each run of the Colored Contraction Algorithm takes polynomial time, and we run it a polynomial number of times.

Let us now demonstrate how this result can be used in order to solve the Minimum Duplication Bipartite problem.

Theorem 3. *The* Minimum Duplication Bipartite *problem is randomized polynomial time solvable when the gene trees are of bounded depth.*

Proof. Remind that, given a binary tree $T=(V,E)$ and a vertex $v \in V$, v^L (resp. v^R) denotes the left (resp. right) child of v and by $\zeta(v)$ the cluster of v *i.e.* the set of all leaves belonging to the subtree rooted in v. Moreover, for ease, ϑ_T is denoting the root of the tree T. Given a gene tree forest $\mathcal{F} = \{T_1 = (V_1, E_1), T_2 = (V_2, E_2), \ldots\}$ built on Λ, considering the definition of the Minimum Duplication Bipartite problem, one wants to define a bipartition (Λ_1, Λ_2) of $\Lambda = \bigcup_{T_i \in \mathcal{F}} V_i$ inducing the minimum number of pre-duplications. In T_i, a node v of V_i is a duplication with respect to (Λ_1, Λ_2),

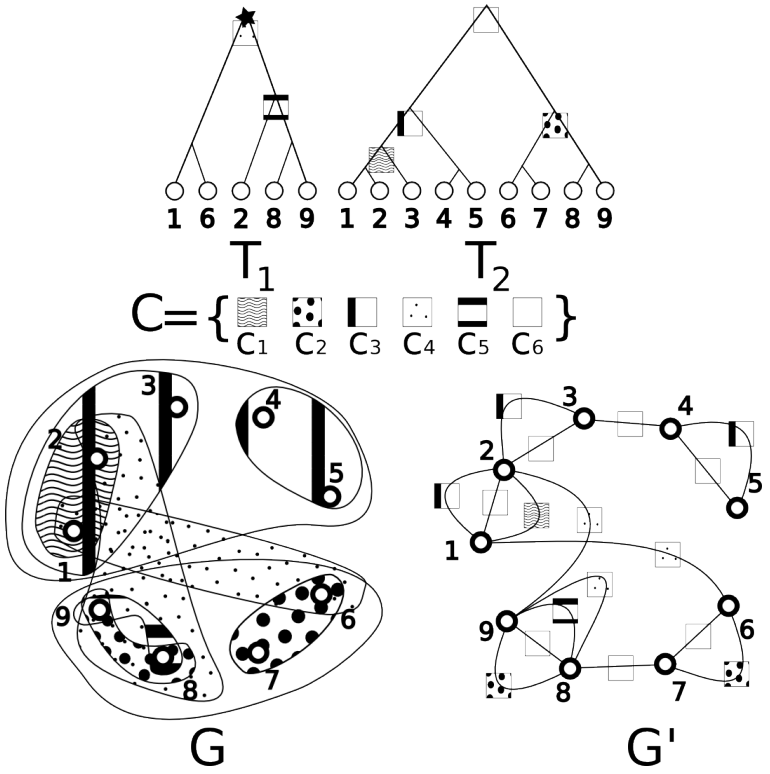

Figure 12: Illustration of the construction of $G_{\mathcal{F}}$ and G' given $\mathcal{F} = (T_1, T_2)$. Considering the minimum colored-cut $\{1, 2, 3, 4, 5\}, \{6, 7, 8, 9\}$ of size 1, the only induced duplication is represented as a star on T_1.

if $\exists v' \in \{v^L, v^R\}$, such that $(\Lambda_1 \bigcap \zeta(v') \neq \emptyset) \wedge (\Lambda_2 \bigcap \zeta(v') \neq \emptyset)$ is true. In other words, v is a duplication if for one of its children – say v' – $\zeta(v')$ contains two leaves not belonging to the same part of the bipartition (Λ_1, Λ_2). Given $\mathcal{F}$ and a set of colors $\mathcal{C}$, we define the following colored hypergraph $G_{\mathcal{F}} = (V, E)$ associated to $\mathcal{F}$. Let $V = \Lambda = \bigcup_{T \in \mathcal{F}} \zeta(\vartheta_T)$ and there are two hyperedges, for any node v_k of the tree T_i, $\alpha_k^i = \{\zeta(v_k^L) : |\zeta(v_k^L)| \geq 2\}$ and $\beta_k^i = \{\zeta(v_k^R) : |\zeta(v_k^R)| \geq 2\}$ colored with color $\texttt{col}(\alpha_k^i) = \texttt{col}(\beta_k^i) = c_k^i \in \mathcal{C}$ in E. An illustration of such construction is provided in Fig. 12. Then in $G_{\mathcal{F}}$, a colored-cut of size k' corresponds to a bipartition of the set Λ inducing k' duplications. Indeed, if the hyperedge α_k^i (resp. β_k^i) belongs to the cut-set, then it induces a duplication for the corresponding vertex v_k in T_i since there exist at least two leaves in $\zeta(v_k^L)$ (resp. $\zeta(v_k^R)$) belonging to different parts of

the bipartition (Λ_1, Λ_2).

Thus, if one can find a minimum colored-cut in such hypergraphs, then one would be able to solve in polynomial time the Minimum Duplication Bipartite problem. Just consider the Colored Contraction Algorithm presented previously in this section. From any colored hypergraph $G_{\mathcal{F}} = (V, E)$, one may build a colored graph $G' = (V, E')$ where any hyperedge $e = \{v_{i1}, v_{i2} \dots v_{ik}\}$ colored with color $c = \texttt{col}(e)$ has been replaced by a path $v_{i1}, v_{i2} \dots v_{ik}$ colored with c in E' (*i.e.* $E' = \{\{v_{ik}, v_{ik+1}\} : v_{ik} \in e, e \in E\}$). Notice that an edge $e \in E'$ colored with c is cut if and only if an hyperedge colored c of $G_{\mathcal{F}}$ is cut. Once this colored graph has been obtained, one may apply the Colored Contraction Algorithm which will produce a minimum colored-cut of G' which also induces a minimum colored cut in $G_{\mathcal{F}}$. Since this algorithm has a complexity exponential in the maximum multiplicity of any color of the considered graph, when the size of each hyperedge is bounded, so does the multiplicity of any color since the maximal size of an hyperedge corresponds to the maximal depth of the input gene trees: leading to a randomized polynomial solution for the Minimum Duplication Bipartite problem. □

5. Conclusion

In this paper we have investigated the complexity of two variants of the Minimum Duplication problem. We have proved that the Minimum Duplication problem is APX-hard, even when the input consists of five uniquely leaf-labelled gene trees. Then, we have shown that the Minimum Duplication Bipartite problem can be solved efficiently by a randomized algorithm when the input gene trees have bounded depth.

A natural open problem is the complexity of the Minimum Duplication Bipartite problem when the gene trees have unbounded depth. Furthermore, it would be interesting to deepen the analysis on the complexity of the Minimum Duplication problem, when the input consists of less than five uniquely leaf-labelled gene trees.

Acknowledgements

The authors acknowledge partial funding from ANR project BIRDS JCJC SIMI 2-2010, and also would like to thanks the anonymous reviewers for valuable arguments and remarks. Paola Bonizzoni and Riccardo Dondi have

been supported by the PRIN 2010/11 grant "Automi e Linguaggi Formali: Aspetti Matematici e Applicativi", code H41J12000190001.

References

[1] G. Blin, P. Bonizzoni, R. Dondi, R. Rizzi, F. Sikora, Complexity insights of the minimum duplication problem, in: M. Bieliková, G. Friedrich, G. Gottlob, S. Katzenbeisser, G. Turán (Eds.), SOFSEM, Vol. 7147 of Lecture Notes in Computer Science, Springer, 2012, pp. 153–164.

[2] J. Felsenstein, Phylogenies from molecular sequences: Inference and reliability, Ann. Review Genet. 22 (1988) 521–565.

[3] E. E. Eichler, D. Sankoff, Structural dynamics of eukaryotic chromosome evolution, Science 301 (5634) (2003) 521–565.

[4] W. M. Fitch, Homology—a personal view on some of the problems, Trends Genet. 16 (2000) 227–231.

[5] L. Arvestad, J. Lagergren, B. Sennblad, The gene evolution model and computing its associated probabilities, J. ACM 56 (2).

[6] L. Arvestad, A.-C. Berglung, J. Lagergren, B. Sennblad, Gene tree reconstruction and orthology analysis based on an integrated model for duplications and sequence evolution., in: D. Gusfield (Ed.), RECOMB 2004, ACM, New York, 2004, pp. 326–335.

[7] P. Bonizzoni, G. Della Vedova, R. Dondi, Reconciling a gene tree to a species tree under the duplication cost model., Theoretical Computer Science 347 (2005) 36–53.

[8] W. Chang, O. Eulenstein, Reconciling gene trees with apparent polytomies, in: D. Chen, D. T. Lee (Eds.), COCOON 2006, Vol. 4112 of LNCS, Heidelberg, 2006, pp. 235–244.

[9] C. Chauve, N. El-Mabrouk, New Perspectives on Gene Family Evolution: Losses in Reconciliation and a Link with Supertrees, in: S. Batzoglou (Ed.), RECOMB, Vol. 5541 of LNCS, Springer, 2009, pp. 46–58.

[10] J. Cotton, R. Page, Rates and patterns of gene duplication and loss in the human genome, Proceedings of the Royal Society of London. Series B 272 (2005) 277–283.

[11] D. Durand, B. Haldórsson, B. Vernot, A hybrid micro-macro-evolutionary approach to gene tree reconstruction, Journal of Computational Biology 13 (2006) 320–335.

[12] B. Ma, M. Li, L. Zhang, From Gene Trees to Species Trees, SIAM J. Comput. 30 (3) (2000) 729–752.

[13] R. Page., Genetree: comparing gene and species phylogenies using reconciled trees., Bioinformatics 14 (1998) 819–820.

[14] R. Page, J. Cotton, Vertebrate phylogenomics: reconciled trees and gene duplications, in: Pacific Symposium on Biocomputing, 2002, pp. 536–547.

[15] J.-P. Doyon, C. Scornavacca, K. Gorbunov, G. Szolloso, V. Ranwez, V. Berry, An effi. algo. for gene/species trees parsim. reconc. with losses, dup. and transf., J. Comp. Biol. 6398 (2010) 93- 108.

[16] M. Hallett, J. Lagergren, A. Tofigh, Simultaneous identification of duplications and lateral transfers, in: RECOMB, ACM, 2004, pp. 347–356.

[17] A. Tofigh, M. Hallett, J. Lagergren, Simultaneous identification of duplications and lateral gene transfers, IEEE/ACM Trans. Comput. Biol. Bioinform. 8 (2011) 517-535.

[18] M. S. Bansal, E. J. Alm, M. Kellis, Efficient algorithms for the reconciliation problem with gene duplication, horizontal transfer and loss, Bioinformatics 28 (12) (2012) 283–291.

[19] M. S. Bansal, E. J. Alm, M. Kellis, Reconciliation revisited: Handling multiple optima when reconciling with duplication, transfer, and loss, Journal of Computational Biology 20 (10) (2013) 738–754.

[20] M. T. Hallett, J. Lagergren, New algorithms for the duplication-loss model, in: RECOMB, ACM, 2000, pp. 138–146.

[21] U. Stege, Gene Trees and Species Trees: The Gene-Duplication Problem is Fixed-Parameter Tractable, in: F. K. H. A. Dehne, A. Gupta, J.-R. Sack, R. Tamassia (Eds.), 6th International Workshop on Algorithms and Data Structures (WADS'99), Vol. 1663 of LNCS, Springer, 1999, pp. 288–293.

[22] M. S. Bansal, R. Shamir, A Note on the Fixed Parameter Tractability of the Gene-Duplication Problem, IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB) 8 (3) (2011) 848–850.

[23] J. Byrka, S. Guillemot, J. Jansson, New results on optimizing rooted triplets consistency, Discrete Appl Math 158 (11) (2010) 1136–1147.

[24] A. Chester, R. Dondi, A. Wirth, Resolving rooted triplet inconsistency by dissolving multigraphs, in: T.-H. H. Chan, L. C. Lau, L. Trevisan (Eds.), TAMC, Vol. 7876 of Lecture Notes in Computer Science, Springer, 2013, pp. 260–271.

[25] M. S. Bansal, J. G. Burleigh, O. Eulenstein, A. Wehe, Heuristics for the Gene-Duplication Problem: A $\Theta(n)$ -Speed-Up for the Local Search, in: T. P. Speed, H. Huang (Eds.), RECOMB, Vol. 4453 of LNCS, Springer, 2007, pp. 238–252.

[26] M. S. Bansal, O. Eulenstein, A. Wehe, The Gene-Duplication Problem: Near-Linear Time Algorithms for NNI-Based Local Searches, IEEE/ACM Trans. Comput. Biology Bioinform. 6 (2) (2009) 221–231.

[27] W.-C. Chang, J. G. B. andDavid F Fernández-Baca, O. Eulenstein, An ILP solution for the gene duplication problem, BMC Bioinformatics (Suppl 1):S14 (12).

[28] A. Ouangraoua, K. M. Swenson, C. Chauve, An Approximation Algorithm for Computing a Parsimonious First Speciation in the Gene Duplication Model, in: E. Tannier (Ed.), RECOMB-CG, Vol. 6398 of LNCS, Springer, Ottawa, Canada, 2010, pp. 290–301.

[29] D. J. A. Welsh, M. B. Powell, An upper bound for the chromatic number of a graph and its application to timetabling problems, The Computer Journal 10 (1) (1967) 85–86.

[30] P. Alimonti, V. Kann, Some APX-completeness results for cubic graphs, Theoretical Comput. Sci. 237 (1–2) (2000) 123–134.

[31] J. Kleinberg, E. Tardos, Algorithm Design, Pearson Education, 2006.